Disk Farm

Distributed farm disk storage

Igor Mandrichenko, CCF, FNAL

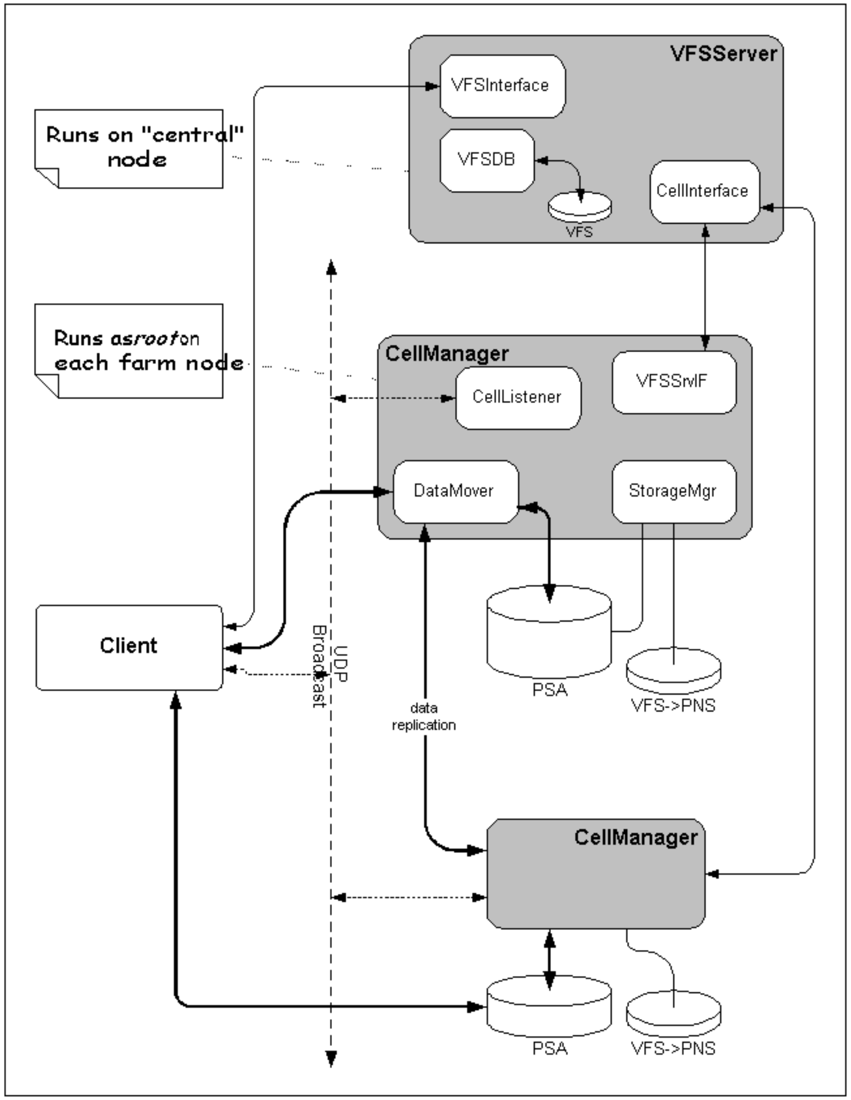
Farm resources

- On central “server” or “head” node
  - Some CPU
  - Big disk
  - Tape drives
  - Network connection to worker nodes
  - Outside network
- On “worker” nodes
  - CPU
  - Network connection to central node(s)
  - Local disk – free or $1/GB
- Local disk on worker nodes seems to be the cheapest but the most difficult to use resource
  - It is distributed over large number of disks*computers
  - Hard to keep track of physical data location
  - Data availability not better than availability of the farm computers

Disk Farm as distributed disk storage management system

- Name space is organized into “virtual file name space”
  - Virtual path: /E123/data/file.5 – this is what user knows
  - Physical path: fnpc221:/local/stage2/XYZ123 – this is what disk farm knows so that user does not have to
- User operates in familiar UNIX-like file name space using familiar commands
- Solution for node unreliability problem: replicate data
  - Make 2,3,4… copies of the file on different nodes
    - Data is easy to reproduce or has short life: 1 copy
    - Data is precious: 2,5,10… copies
  - Disk Farm replicates data off-line
- Load sharing and control
  - Each node has a limit for the number of simultaneous reads/writes
  - Load is evenly distributed and optimized

Disk Farm design



User interface

- Basic UNIX file system commands operate in virtual file name space
  - `dfarm **ls** <path>|<wildcard>`
  - `dfarm **mkdir** <vpath>`
  - `dfarm **rmdir** <vpath>`

  -
    ```
    dfarm put [-v] [-t <timeout>]
              [-n <ncopies>] <local path> <vpath>
    ```
  - `dfarm **get** [-v] [-t <timeout>] <vpath> <local path>`
  - `dfarm **rm** <vpath>|<wildcard>`
  - `dfarm **ln** <vpath> <local path>`
- Additional commands
  - `dfarm **info** <vpath>`
    - Prints where the file is stored
  - `dfarm **ping**`
    - Prints list of available disk farm nodes and their load (response time, transactions)
  - `dfarm **stat** <node>`
    - Prints status of individual farm node (disk space availability)

User Interface (continued)

- File/directory attributes
  - User can define arbitrary attributes for files and directories and assign them arbitrary text values
  - Attributes may be inherited from a directory down
  - User can list files/directory with certain attributes
- Administrative functions
  - File re-replication
    - Make additional replicas of an existing file
  - Node hold/release
    - Held node is read-only
  - Node replication
    - Node is about to go down, copy data stored there to other nodes

Local and remote access

- Local access through proprietary interface, UI, Python API
  - Fast
  - Scalable
  - Less secure, security schema is similar to NFS

- Remote access through Kerberized or GSI (Grid) FTP interface
  - Still fast and scalable (data does not go through central node)
  - Strong security
  - Third party transfers allow direct cluster-to-cluster transfers

What it can be used for

- Three types of storage
  - Scratch – “sandbox” created for batch process
  - Temporary – data is “parked” between production phases or analisys iterations and is to be deleted
  - Permanent – data is stored indefinitely

- Disk Farm perfect as temporary storage
  - Not permanent: replication helps data live longer, but still not infinitely
  - Permanent or temporary largely depends on local support policies

- Remote (GridFTP) interface can be used to expose a cluster as a grid storage element

- Data can be pre-staged to a cluster before jobs start there

Existing installations

- CDF production farm
  - ~150 nodes
  - 16TB capacity, 10TB used
  - ~10 TB/day transferred to/from/inside
  - “single” user
- Fixed Target farm
  - 90 nodes
  - 2TB capacity, ~1TB used
  - Multiple users
- D0 production farm
  - 340 nodes
  - ~5TB, 0.3TB used
  - “single” user